ReCOP Instruction Set Revision MAY2015

Release Note

This ReCOP specification revision is primarily aimed at implementation that will be used in TP-JOP processor in COMPSYS701 course. This document should be read in conjunction with [1].

Instruction	Description	Register Transfers	Addressing Modes			
			Inherent	Immediate	Direct	Register
AND Rz Rx #Operand	The contents of Rx and Rz / Operand are ANDed and the result is stored in Rz	Rz <- Rx AND Operand		X		
AND Rz Rz Rx		Rz <- Rz AND Rx				X
OR Rz Rx #Operand	The contents of Rx and Rz / Operand are ORed and the result is stored in Rz	Rz <- Rx OR Operand		X		
OR Rz Rz Rx		Rz <- Rz OR Rx				X
ADD Rz Rx #Operand	The contents of Rx and Rz / Operand are added and the result is stored in Rz	Rz <- Rx + Operand		X		
ADD Rz Rz Rx		Rz <- Rz + Rx				X
SUBV Rz Rx #Operand	The contents of Rx and Rz / Operand are subtracted and the result is stored in Rz	Rz <- Rx - Operand		X		
SUB Rz #Operand	The contents of Rz and the Operand are subtracted but the result is not stored	Rz - Operand		X		
LDR Rz #Operand	Load Rz with the content of immediate value / memory location pointed to by Rx or Operand	Rz <- Operand		X		
LDR Rz Rx		Rz <- M[Rx]				X
LDR Rz $Operand		Rz <- M[Operand]			X	
STR Rz #Operand	Store the content of Rx / immediate value, into memory location pointed to by Rz / direct address	M[Rz] <- Operand		X		
STR Rz Rx		M[Rz] <- Rx				X
STR Rx $Operand		M[Operand] <- Rx			X	
JMP #Operand	Jump to address location unconditionally	PC <- Operand		X		
JMP Rx		PC <- Rx				X
PRESENT Rz #Operand	Jump to address location if the thread pointed to by Rz is not present else continue execution	if Rz(15..0)=0x0000 then PC<-Operand else NEXT		X		
DCALLBL Rx	Store the content of R7 and Rx in DPCR register	DPCR <- Rx & R7				X
DCALLBL Rx #Operand	Store the content of Rx and Operand in DPCR register	DPCR <- Rx & Operand		X		
DCALLNB Rx	Store the content of R7 and Rx in DPCR register	DPCR <- Rx & R7				X
DCALLNB Rx #Operand	Store the content of Rx and Operand in DPCR register	DPCR <- Rx & Operand		X		
SZ Operand	Jump to address location if Z=1 else continue execution	if Z=1 then PC <- Operand else NEXT		X		
CLFZ	Clear Zero flag	Z <- 0	X			
CER	Clear Environment Ready bit	ER <- 0	X			
CEOT	Clear EOT bit	EOT <- 0	X			
SEOT	Set EOT bit	EOT <- 1	X			
LER Rz	The content of ER is stored in Rz	Rz <- ER				X
SSVOP Rx	Load SVOP with the content of Rx	SVOP <- Rx				X

LSIP Rz	Load Rz with the content of SIP	Rz <- SIP				X
SSOP Rx	Load SOP with the content of Rx	SOP <- Rx				X
NOOP	No operation		X			
MAX Rz #Operand	Compare two 16-bit nums and store larger one into Rz	Rz <- MAX{Rz, #Operand}		X		
STRPC $Operand	Stores program counter into the specified memory address	M[Operand] <- PC			X	

Instruction	Addressing Mode	Instruction Operations
AND Rz Rx #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- Rx AND Operand
AND Rz Rz Rx	~~Register~~	T1 : No Operation T2 : Rz <- Rz AND Rx
OR Rz Rx #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- Rx OR Operand
OR Rz Rz Rx	~~Register~~	T1 : No Operation T2 : Rz <- Rz OR Rx
ADD Rz Rx #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- Rx + Operand
ADD Rz Rz Rx	~~Register~~	T1 : No Operation T2 : Rz <- Rz + Rx
SUBV Rz Rx #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- Rx - Operand
SUB Rz #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz - Operand
LDR Rz #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- Operand
LDR Rz Rx	~~Register~~	T1 : No Operation T2 : Rz <- M[Rx]
LDR Rz $Operand	~~Direct~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- M[Operand]
STR Rz #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : M[Rz] <- Operand
STR Rz Rx	~~Register~~	T1 : No Operation T2 : M[Rz] <- Rx
STR Rx $Operand	~~Direct~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : M[Operand] <- Rx
JMP #Operand	~~Immediate~~	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : PC <- Operand

JMP Rx	Register	T1 : No Operation T2 : PC <- Rx
PRESENT Rz #Operand	Immediate	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : if Rz(15..0)=0x0000 then PC<-Operand else NEXT
DCALLBL Rx	Register	T1 : No Operation T2 : DPCR <- R7 & Rx
DCALLBL Rx #Operand	Immediate	T1 : No Operation T2 : DPCR <- Rx & Operand
DCALLNB Rx	Register	T1 : No Operation T2 : DPCR <- R7 & Rx
DCALLNB Rx #Operand	Immediate	T1 : No Operation T2 : DPCR <- Rx & Operand
SZ #Operand	Immediate	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : if Z=1 then PC <- Operand else NEXT
CLFZ	Inherent	T1 : No Operation T2 : Z <- 0
CER	Inherent	T1 : No Operation T2 : ER <- 0
CEOT	Inherent	T1 : No Operation T2 : EOT <- 0
SEOT	Inherent	T1 : No Operation T2 : EOT <- 1
LER Rz	Register	T1 : ld_er <- '1' T2 : Rz <- ER
SSVOP Rx	Register	T1 : No Operation T2 : SVOP <- Rx
LSIP Rz	Register	T1 : ld_sip <- '1' T2 : Rz <- SIP
SSOP Rx	Register	T1 : No Operation T2 : SOP <- Rx
NOOP	Inherent	T1 : No Operation T2 : No Operation
MAX Rz #Operand	Immediate	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : Rz <- MAX{ Rz, #Operand}
STRPC $Operand	Direct	T1 : IR(15..0) <- M[PC] , PC <- PC+1 T2 : M[Operand] <- PC

Instruction	Opcode
AND	001000
OR	001100
ADD	111000
SUBV	000011
SUB	000100
LDR	000000
STR	000010
JMP	011000
PRESENT	011100
DCALLBL	101000
DCALLNB	101001
SZ	010100
CLFZ	010000
CER	111100
CEOT	111110
SEOT	111111
LER	110110
SSVOP	111011
LSIP	110111
SSOP	111010
NOOP	110100
MAX	011110
STRPC	011101

Addressing mode	bits
Inherent	00
Immediate	01
Direct	10
Register	11

IR

AM (2)	Opcode(6)	Rz(4)	Rx(4)	Operand(16)

References:
[1] Z. Salcic: “ReCOP, Reactive Coprocessor for Tandem Processor – Embedded Execution Platform for SystemJ Language”, Technical report, Embedded Systems Lab, University of Auckland, February 2014